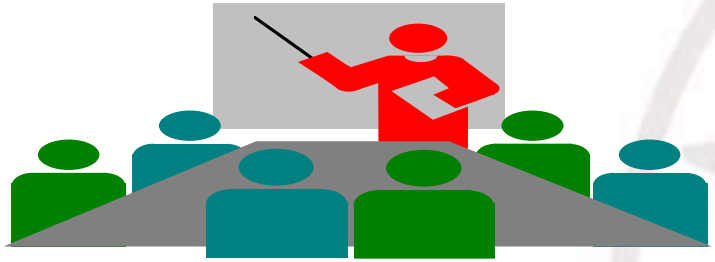


逻辑与计算机设计基础实验与课程设计

实验十二

移位寄存器设计与应用

施青松
Asso. Prof. Shi Qingsong
College of Computer Science and Technology, Zhejiang University
zjsqs@zju.edu.cn

Course Outline

- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

实验目的

1. 掌握移位寄存器的工作原理及设计方法；

2. 掌握串、并数据转换的概念与方法；

3. 掌握同步串行传输方法

实验环境

实验设备

1. 计算机（Intel Core i3以上，1GB内存以上）系统

2. SWORD Kintex™-7 Board开发板

3. Xilinx ISE12.4及以上开发工具

材料

无

Course Outline

- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

实验任务

1. 结构化描述方法设计32位带并行输入的右移移位寄存器；

2. 利用该移位寄存器实现一个64位并行－串行转换器；

3. 设计32位移位寄存器加入实验十的混合计算器中

Course Outline

- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

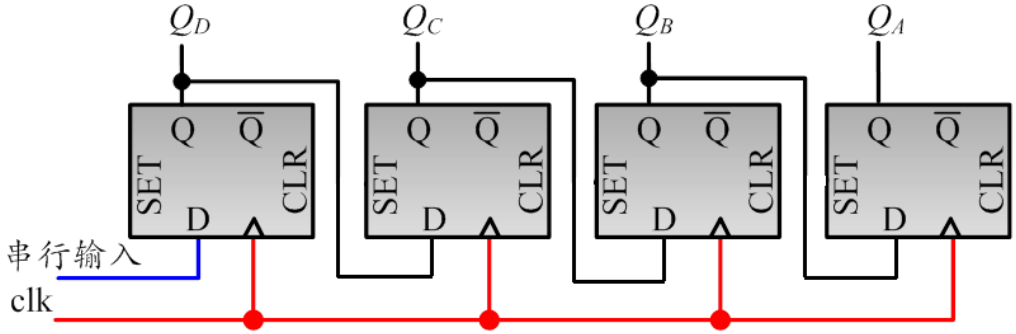
移位寄存器

◎ 移位寄存器

- 每来一个时钟脉冲，寄存器中的数据按顺序移动一位
  - ⊙数据移动方式：左移、右移
- 必须采用主从触发器或边沿触发器
- 不能采用电平触发器

◎ 数据输入输出方式

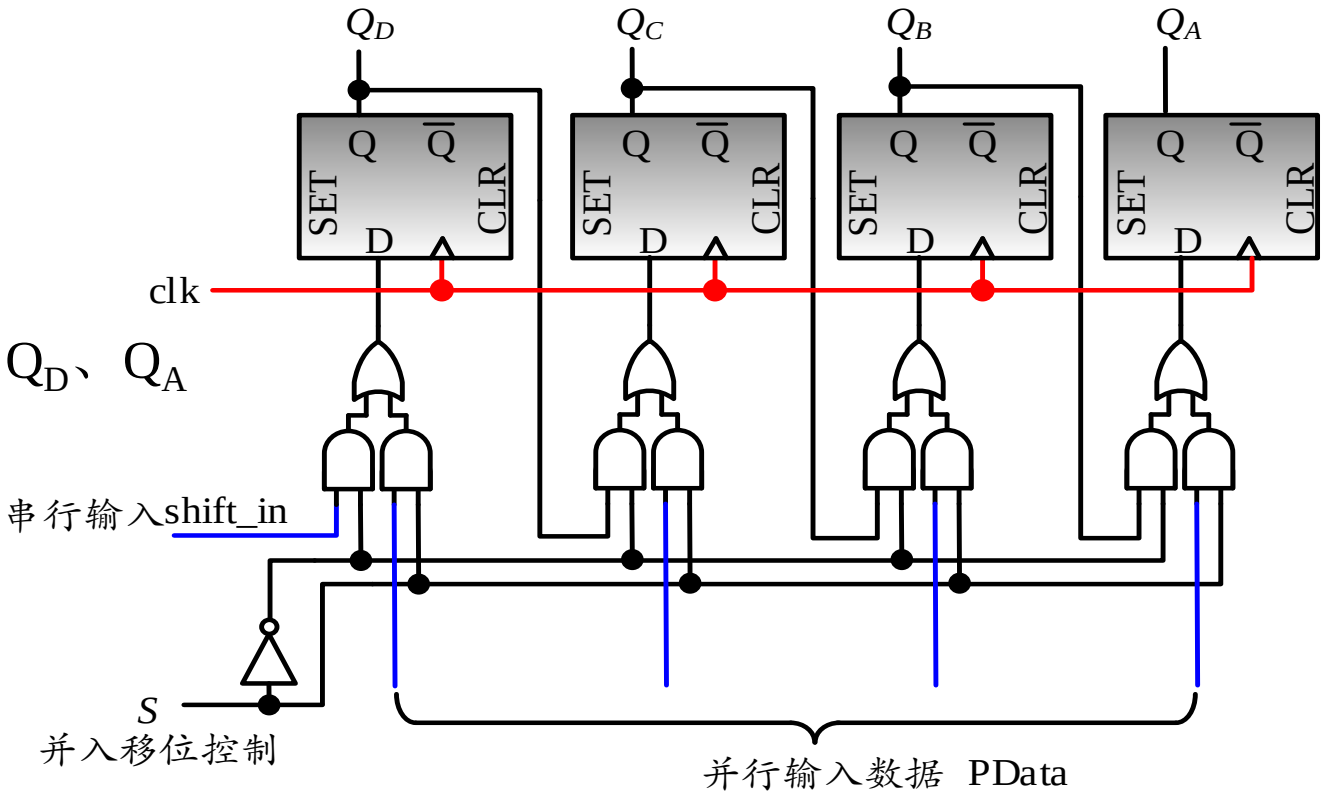
- 串行输入，串行输出
- **串行输入，并行输出**
- **并行输入，串行输出**



D触发器

带并行置入的右移移位寄存器

- ◎ 输入
  - 串行输入
    - Shift_in
  - 并行输入
    - PData
- ◎ 输出
  - 并行输出
    - Q_D、Q_C、Q_D、Q_A
  - 串行输出
    - Q_A



双向通用移位寄存器

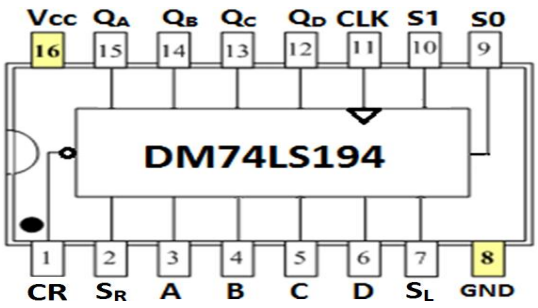
◎ DM/SN74LS194：

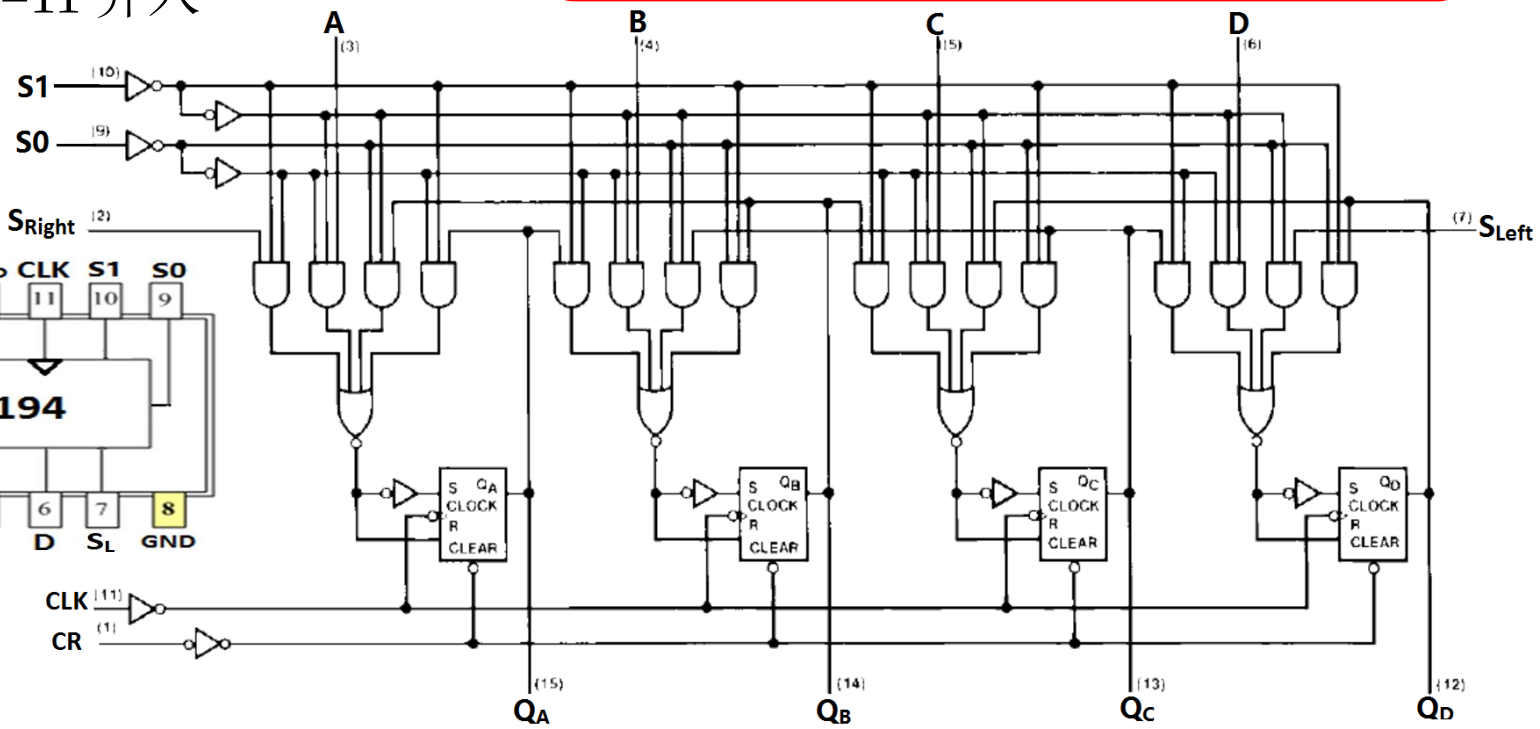
双向通用移位寄存器

- 可用于并串转换

S1S0=00 保持
S1S0=01 右移
S1S0=10 左移
S1S0=11 并入

Inputs										Outputs			
CR	Mode		Clock	Serial		Parallel				Q_A	Q_B	Q_C	Q_D
	S1	S0		S_{Left}	S_{Right}	A	B	C	D				
L	X	X	X	X	X	X	X	X	X	L	L	L	L
H	X	X	L	X	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}
H	H	H	↑	X	X	a	b	c	d	a	b	c	d
H	L	H	↑	X	H	X	X	X	X	H	Q_{An}	Q_{Bn}	Q_{Cn}
H	L	H	↑	X	L	X	X	X	X	L	Q_{An}	Q_{Bn}	Q_{Cn}
H	H	L	↑	H	X	X	X	X	X	Q_{Bn}	Q_{Cn}	Q_{Dn}	H
H	H	L	↑	L	X	X	X	X	X	Q_{Bn}	Q_{Cn}	Q_{Dn}	L
H	L	L	X	X	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}



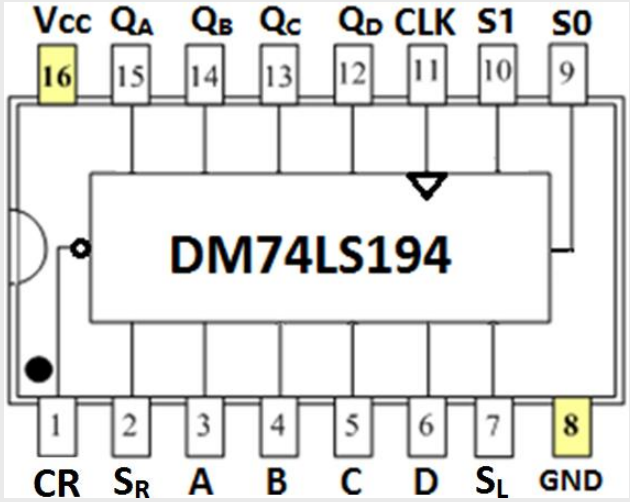


浙江大学 系
ZheJiang University

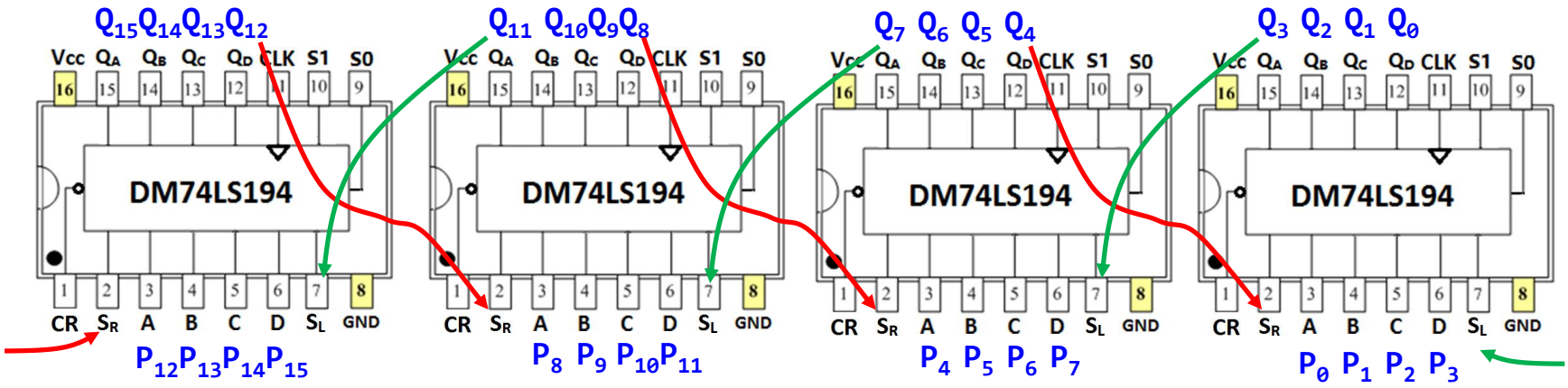
原语参考描述：4位通用移位寄存器

```
module DM74LS194(clk,CR,S1,S0,A,B,C,D, SL, SR, QA, QB, QC, QD);
// ……
    INV   GS0(.I(S0), .O(nS0)),
          GS1(.I(S1), .O(nS1));

    MB_DFF Shift0(.Cp(clk), .D(D0), .Rn(CR), .Sn(1'b1), .Q(QA), .Qn()),
           Shift1(.Cp(clk), .D(D1), .Rn(CR), .Sn(1'b1), .Q(QB), .Qn()),
           Shift2(.Cp(clk), .D(D2), .Rn(CR), .Sn(1'b1), .Q(QC), .Qn()),
           Shift3(.Cp(clk), .D(D3), .Rn(CR), .Sn(1'b1), .Q(QD), .Qn());

    OR4   GD0(.I0(HD0), .I1(RD0), .I2(LD0), .I3(PD0), .O(D0)),
          GD1(.I0(???), .I1(???), .I2(???), .I3(???), .O(D1)),
          GD2(.I0(???), .I1(???), .I2(???), .I3(???), .O(D2)),
          GD3(.I0(???), .I1(???), .I2(???), .I3(???), .O(D3));

    AND3  GH0(.I0(nS1), .I1(nS0), .I2(QA),  .O(HD0)),
          GH1(.I0(nS1), .I1(nS0), .I2(???), .O(HD1)),
          GH2(.I0(nS1), .I1(nS0), .I2(???), .O(HD2)),
          GH3(.I0(nS1), .I1(nS0), .I2(???), .O(HD3));

    AND3  SR0(.I0(nS1), .I1(S0),  .I2(???),  .O(RD0)),
          SR1(.I0(nS1), .I1(S0),  .I2(???), .O(RD1)),
          SR2(.I0(nS1), .I1(S0),  .I2(???), .O(RD2)),
          SR3(.I0(nS1), .I1(S0),  .I2(SR), .O(RD3));

    AND3  SL0(.I0(S1), .I1(nS0),  .I2(SL), .O(LD0)),
          SL1(.I0(S1), .I1(nS0),  .I2(???), .O(LD1)),
          SL2(.I0(S1), .I1(nS0),  .I2(???), .O(LD2)),
          SL3(.I0(S1), .I1(nS0),  .I2(???), .O(LD3));

    AND3  P0(.I0(S1), .I1(S0),  .I2(A), .O(PD0)),
          P1(.I0(S1), .I1(S0),  .I2(???), .O(???)),
          P2(.I0(S1), .I1(S0),  .I2(???), .O(???)),
          P3(.I0(S1), .I1(S0),  .I2(???), .O(???));
endmodule
```

左右相反，仅供参考



双向通用移位寄存器扩展



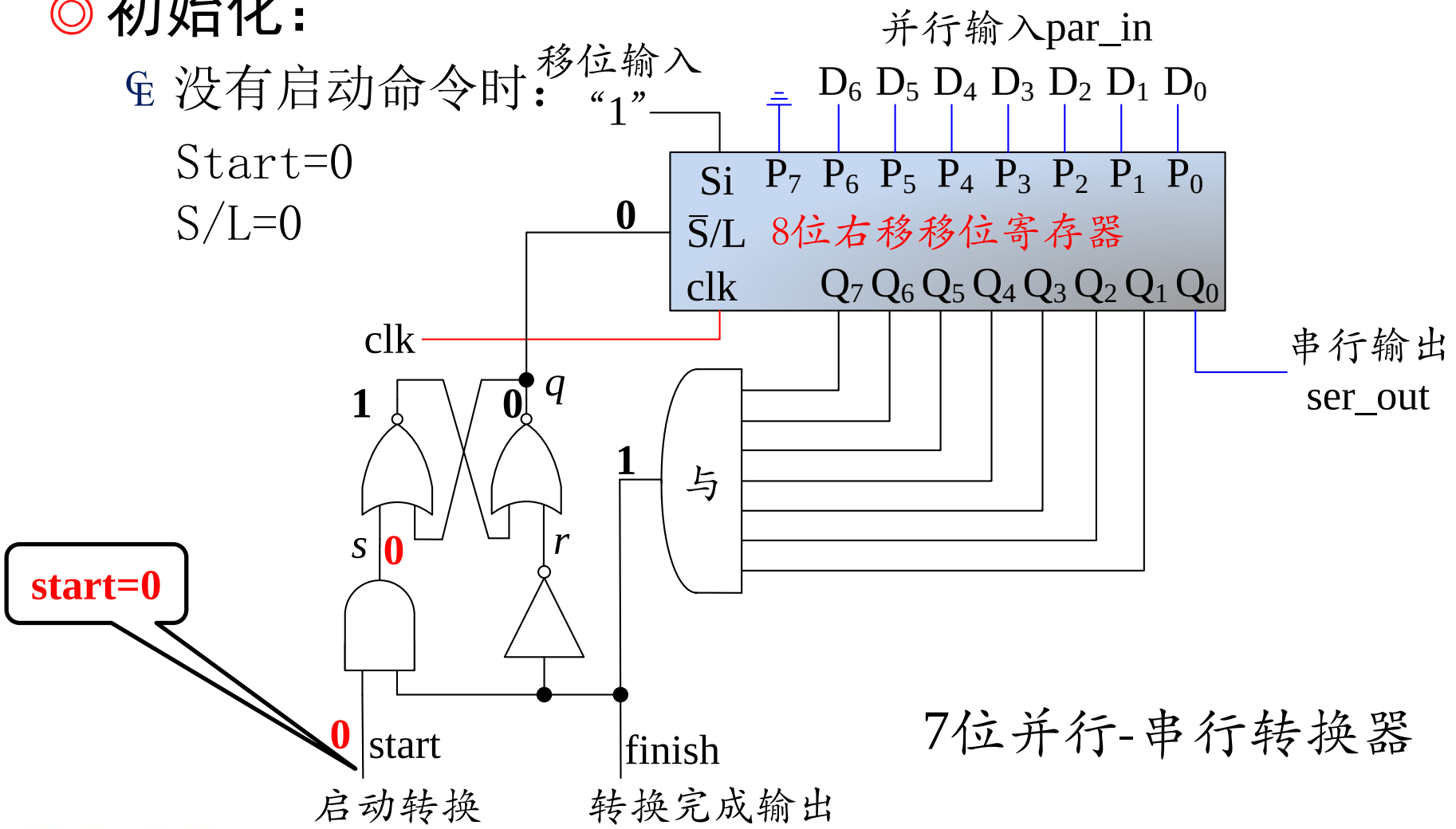
32位通用移位寄存器扩展描述

```
module shift_reg_32bit(clk, clear, S1,S0, SL,SR, PData, Q );
//……
wire CR = ~clear;
    DM74LS194   SH0(.clk(clk), .CR(CR), .S1(S1), .S0(S0), .SL(SL), .SR(Q[4]),
                    .A(PData[0]), .B(PData[1]), .C(PData[2]), .D(PData[3]),
                    .QA(Q[0]),    .QB(Q[1]),    .QC(Q[2]),    .QD(Q[3])),
                SH1(.clk(clk), .CR(CR), .S1(S1), .S0(S0), .SL(???), .SR(???),
                    .A(PData[4]), .B(PData[5]), .C(PData[6]), .D(PData[7]),
                    .QA(Q[4]),    .QB(Q[5]),    .QC(Q[6]),    .QD(Q[7])),
                SH2(.clk(clk), .CR(CR), .S1(S1), .S0(S0), .SL(Q???), .SR(???),
                    .A(PData[8]),.B(PData[9]),.C(PData[10]),.D(PData[11]),
                    .QA(Q[8]),   .QB(Q[9]),   .QC(Q[10]),   .QD(Q[11])),
                SH3(.clk(clk), .CR(CR), .S1(S1), .S0(S0), .SL(???), .SR(???),
                    .A(PData[12]),.B(PData[13]),.C(PData[14]),.D(PData[15]),
                    .QA(Q[12]),   .QB(Q[13]),   .QC(Q[14]),   .QD(Q[15]));
    DM74LS194   SH4(.clk(clk), .CR(CR), .S1(S1), .S0(S0), .SL(Q???), .SR(???),
                    .A(PData[16]),.B(PData[17]),.C(PData[18]),.D(PData[19]),
                    .QA(Q[16]),   .QB(Q[17]),   .QC(Q[18]),   .QD(Q[19])),
                SH5(.clk(clk), .CR(CR), .S1(S1), .S0(S0), .SL(???), .SR(???),
                    .A(PData[20]),.B(PData[21]),.C(PData[22]),.D(PData[23]),
                    .QA(Q[20]),   .QB(Q[21]),   .QC(Q[22]),   .QD(Q[23])),
                SH6(.clk(clk), .CR(CR), .S1(S1), .S0(S0), .SL(???), .SR(???),
                    .A(PData[24]),.B(PData[25]),.C(PData[26]),.D(PData[27]),
                    .QA(Q[24]),   .QB(Q[25]),   .QC(Q[26]),   .QD(Q[27])),
                SH7(.clk(clk), .CR(CR), .S1(S1), .S0(S0), .SL(Q[27]), .SR(SR),
                    .A(PData[28]),.B(PData[29]),.C(PData[30]),.D(PData[31]),
                    .QA(Q[28]),   .QB(Q[29]),   .QC(Q[30]),   .QD(Q[31]));
endmodule
```

同步串行传输：并-串转换器-初始

◎ 初始化：

- 没有启动命令时：
  Start=0
  S/L=0

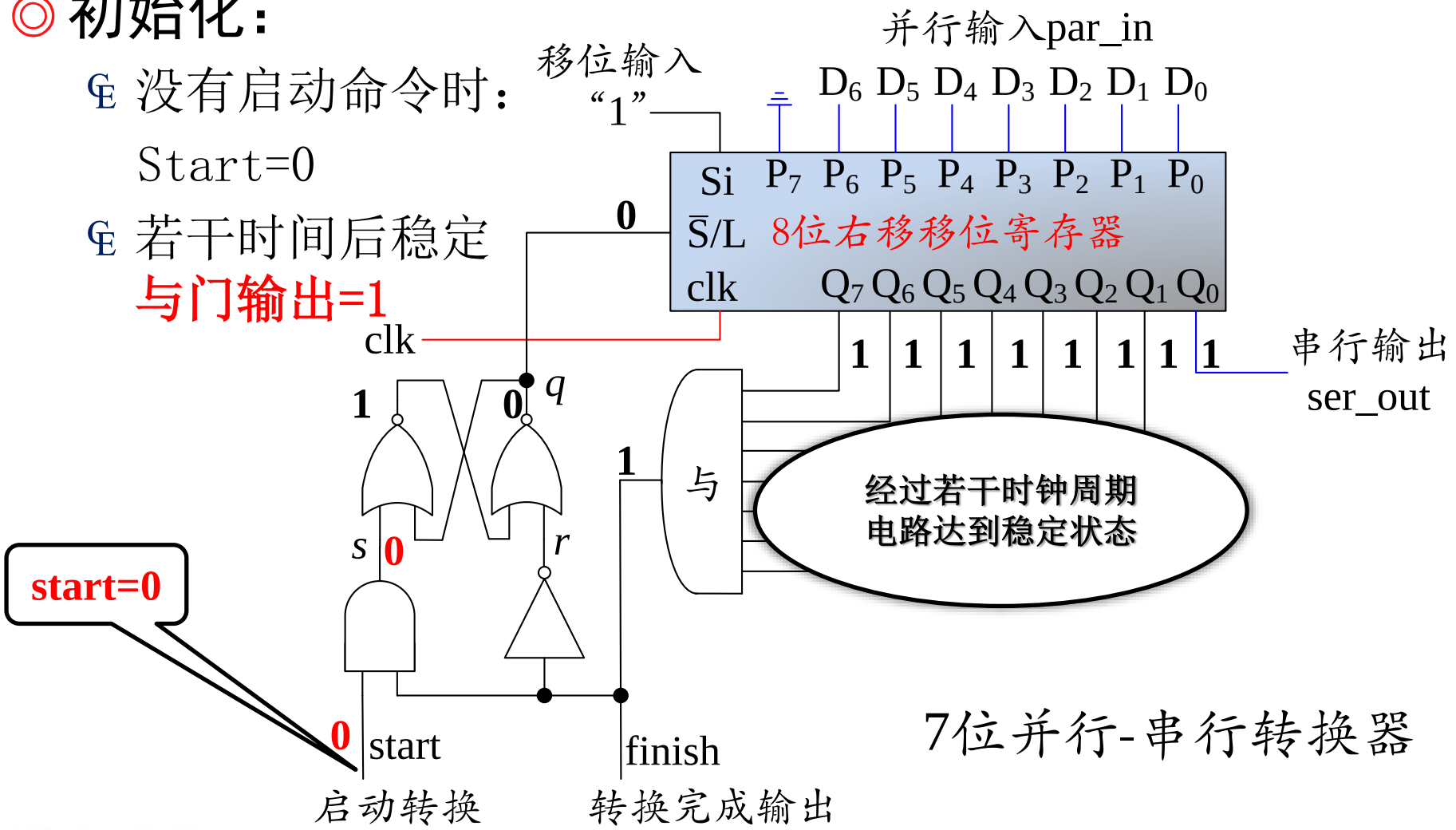


7位并行-串行转换器

同步串行传输：并-串转换器-初始

◎ 初始化：

- 没有启动命令时：
  Start=0
- 若干时间后稳定
  与门输出=1



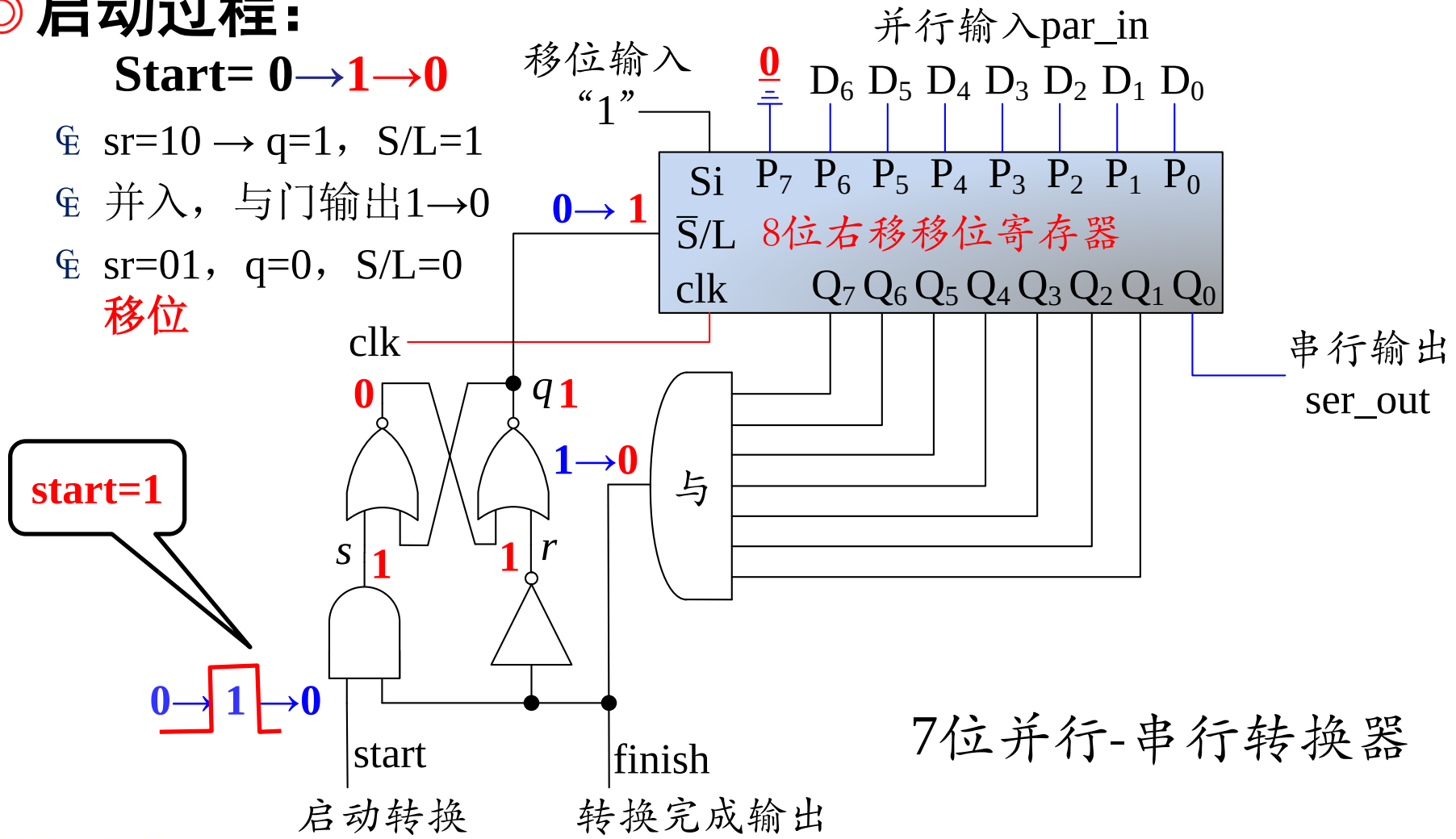
7位并行-串行转换器

同步串行传输：并-串转换器-转换启动

◎ 启动过程：

Start= 0→1→0

- sr=10 → q=1，S/L=1
- 并入，与门输出1→0
- sr=01，q=0，S/L=0
  移位

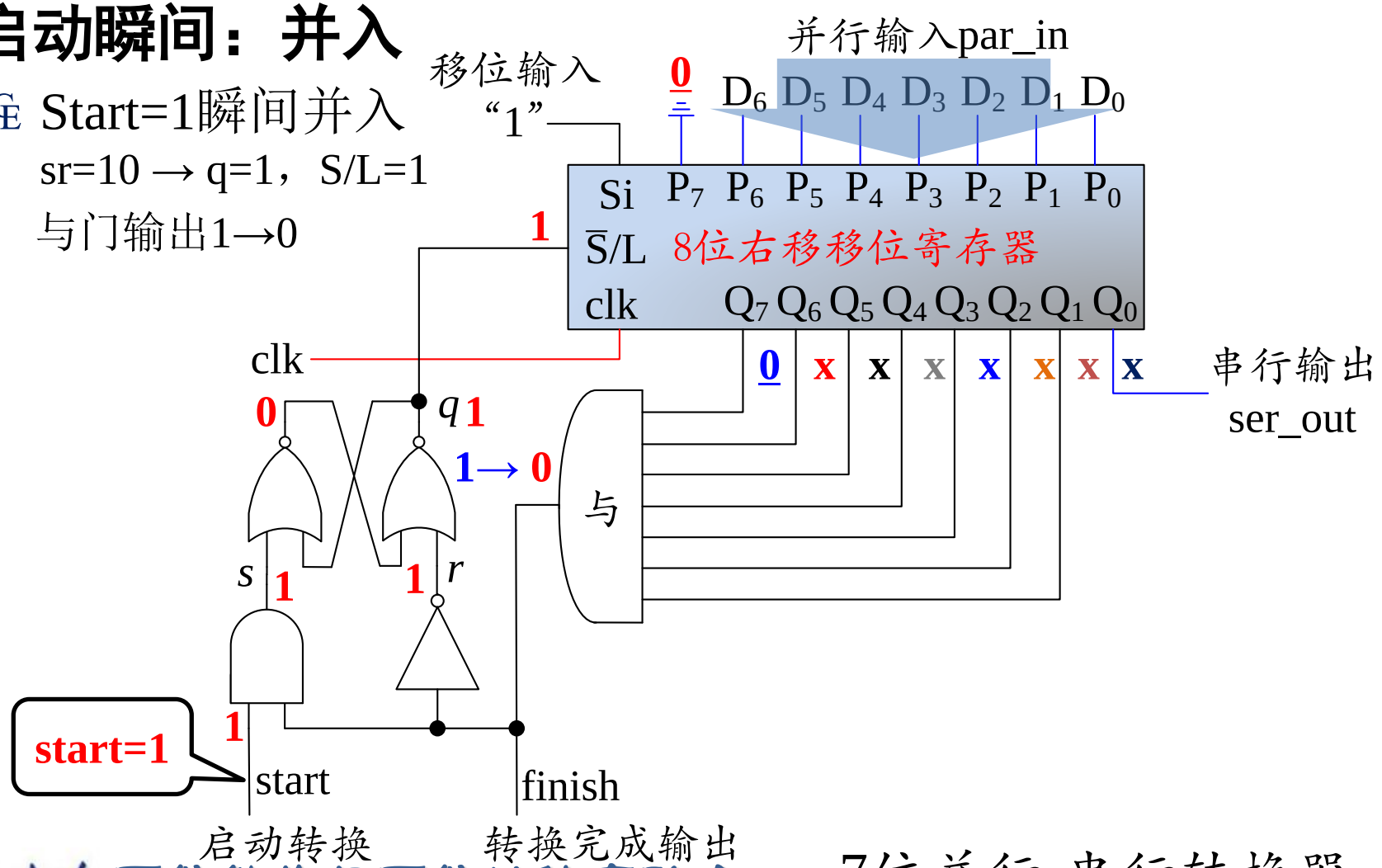


7位并行-串行转换器

同步串行传输：并-串转换器-并入

◎ 启动瞬间：并入

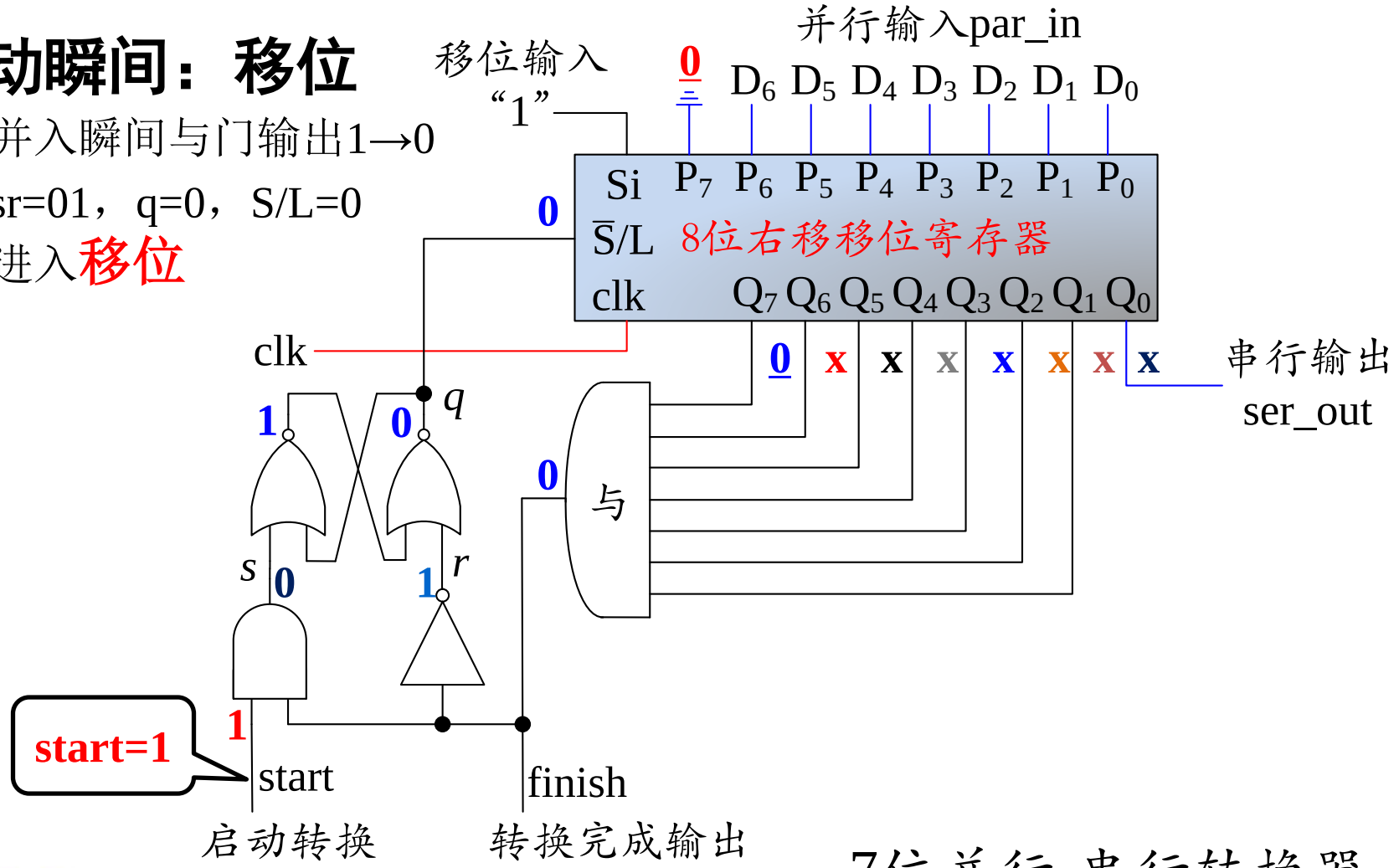
- Start=1瞬间并入
  sr=10 → q=1，S/L=1
  与门输出1→0



并行—串行转换器：转换-进入移位

◎ 启动瞬间：移位

- 并入瞬间与门输出1→0
- sr=01，q=0，S/L=0
  进入**移位**

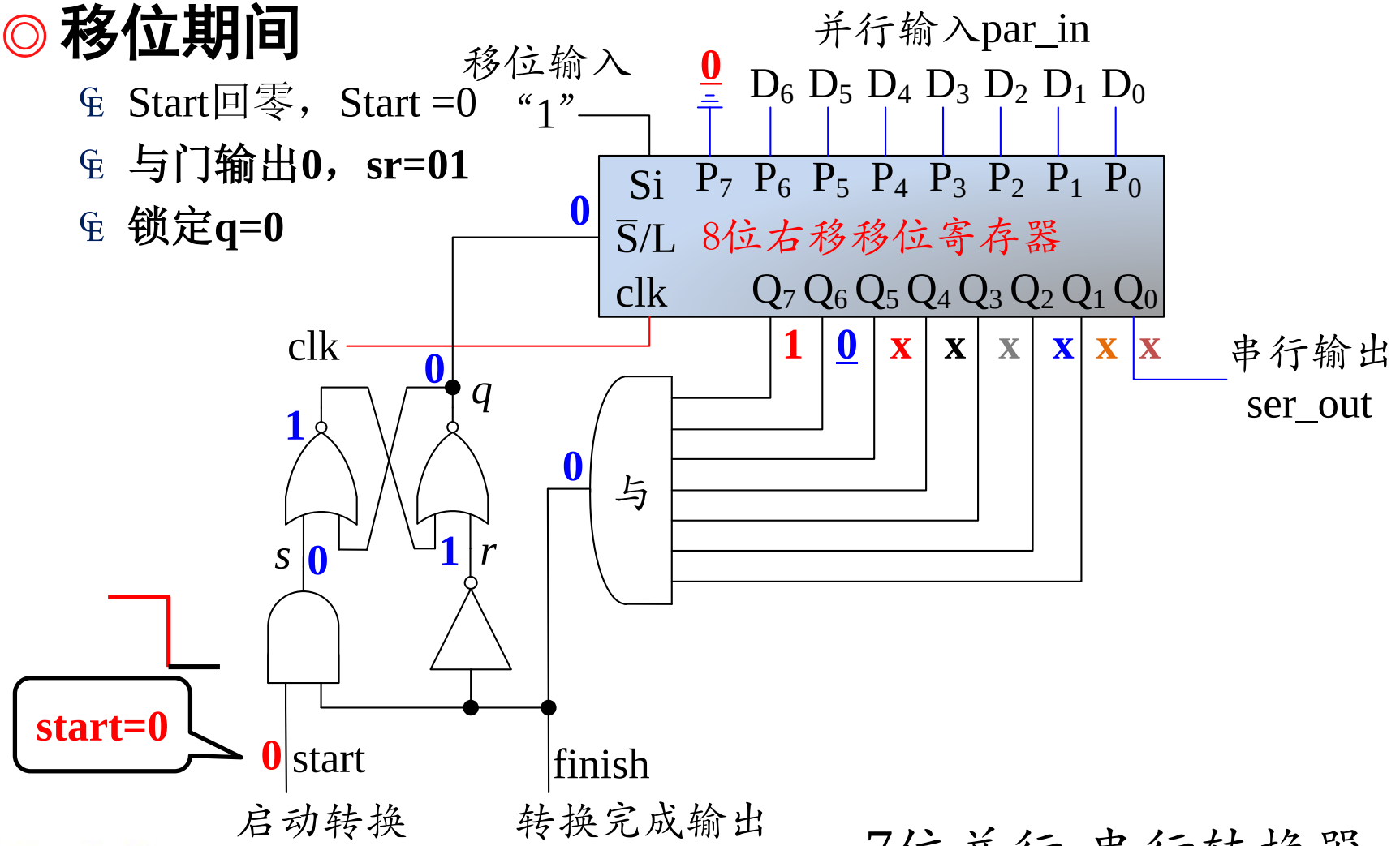


7位并行-串行转换器

并行—串行转换器：传输-移位

◎ 移位期间

- Start回零，Start =0
- 与门输出0，sr=01
- 锁定q=0

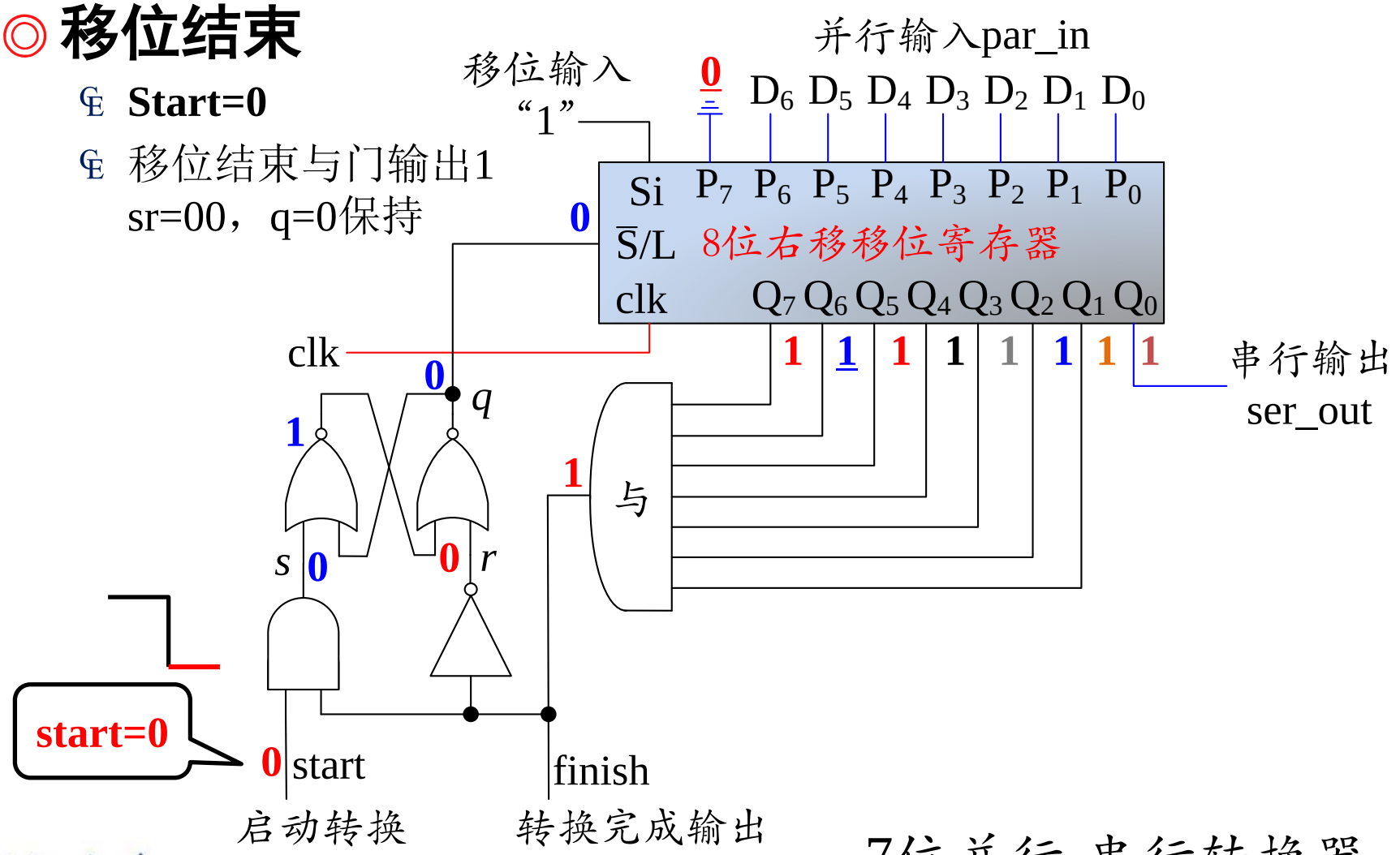


7位并行-串行转换器

并行—串行转换器：转换–结束

◎ 移位结束

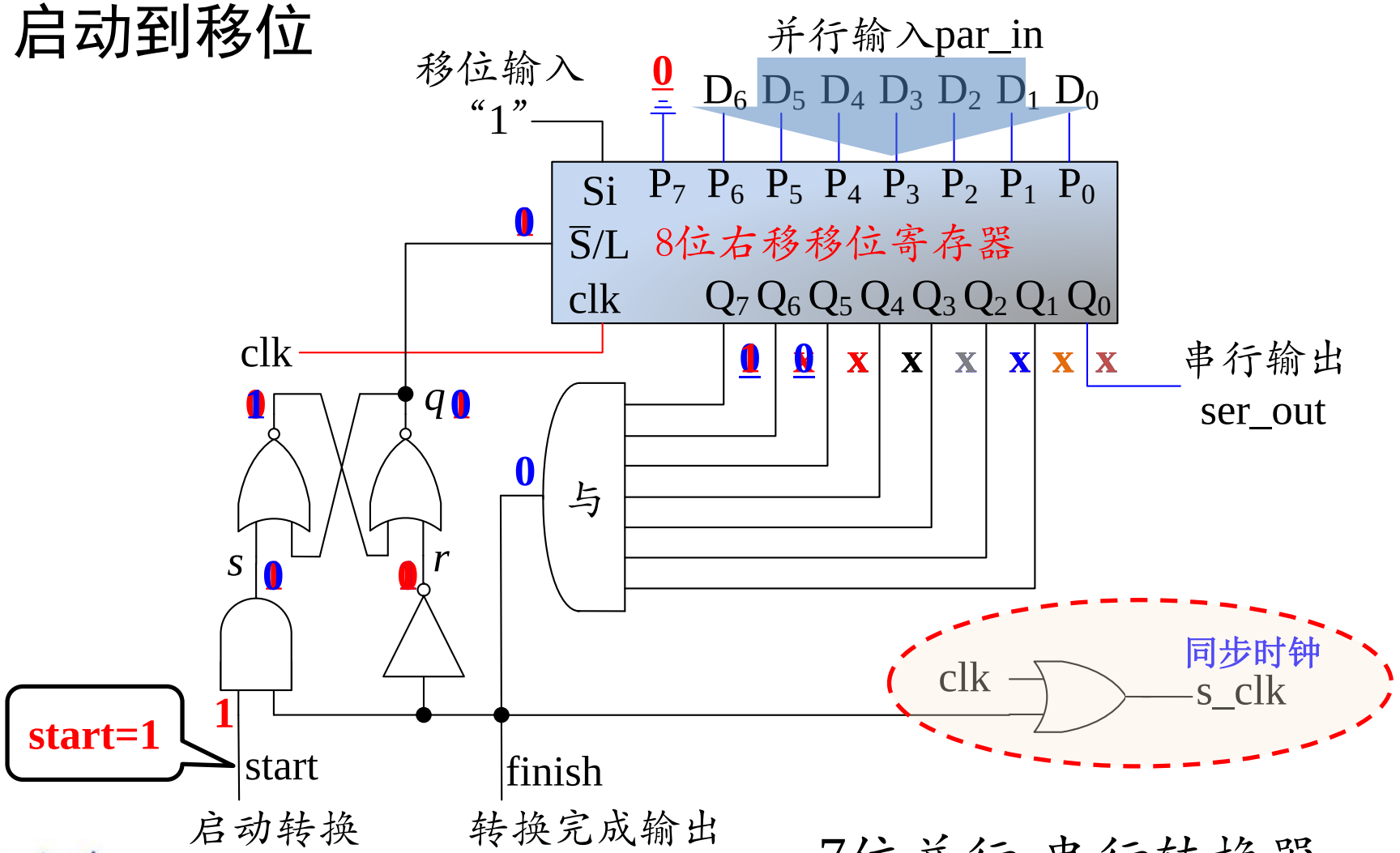
- Start=0
- 移位结束与门输出1
  sr=00，q=0保持



7位并行-串行转换器

并行—串行转换器：转换-连续过程

◎ 启动到移位



7位并行-串行转换器

总结：并行－串行转换器工作过程

- **并行－串行转换器设计原理**
  - 当没有启动命令（低电平）时，电路上电后经过若干个时钟脉冲后将会稳定在RS触发器输出q＝0，移位寄存器Q_7-Q_0＝11111111的状态。
  - 当启动命令（高电平）加至启动输入端时，RS触发器的输出端q被置1，7位并行数据及标志码“0”在第一个clk的作用下同时置入移位寄存器。

    此时，由于P_7=0导致Q_7=0，使得七输入与门的输出变成0，
    - 一方面封锁启动命令的输入，
    - 另一方面通过非门在RS的触发器的r端输入1，使RS触发器的输出q=0，移位寄存器进入移位状态。
    - 再在时钟脉冲作用下，一方面使并行数据串行移出，另一方面又不断将“1”移入寄存器。
    - 等第7个脉冲来到后，七输入与门的输入已全为“1”，使得其输出变为1，标志着转换完成，同时解除对启动信号的封锁
    - 当再来一个启动命令时又可以再次进行并行－串行转换

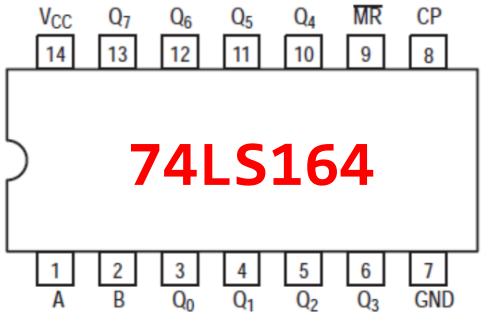
7位并串行转换参考描述：结构描述

```
module transfer(clk, start, par_in, finish, ser_out);
    input clk;
    input start;
    input [6:0] par_in;
    output finish;
    output ser_out;
    wire clk, start, finish, ser_out;
    wire [6:0] par_in;
    wire [7:0] q;
    wire s, r, rs_q, rs_nq;
    shift_reg M_sr (clk, rs_q, 1'b1, {1'b0, par_in[6:0]}, q);
    NOR2   RS_LATCH_1 (.I0(r), .I1(rs_nq), .O(rs_q)),
           RS_LATCH_2 (.I0(s), .I1(rs_q), .O(rs_nq));        // RS Latch
    AND2   s_1 (.I0(start), .I1(finish), .O(s));
    INV    r_1 (.I(finish), .O(r));
    assign finish = &q[7:2];
    assign ser_out = q[0];
endmodule
```

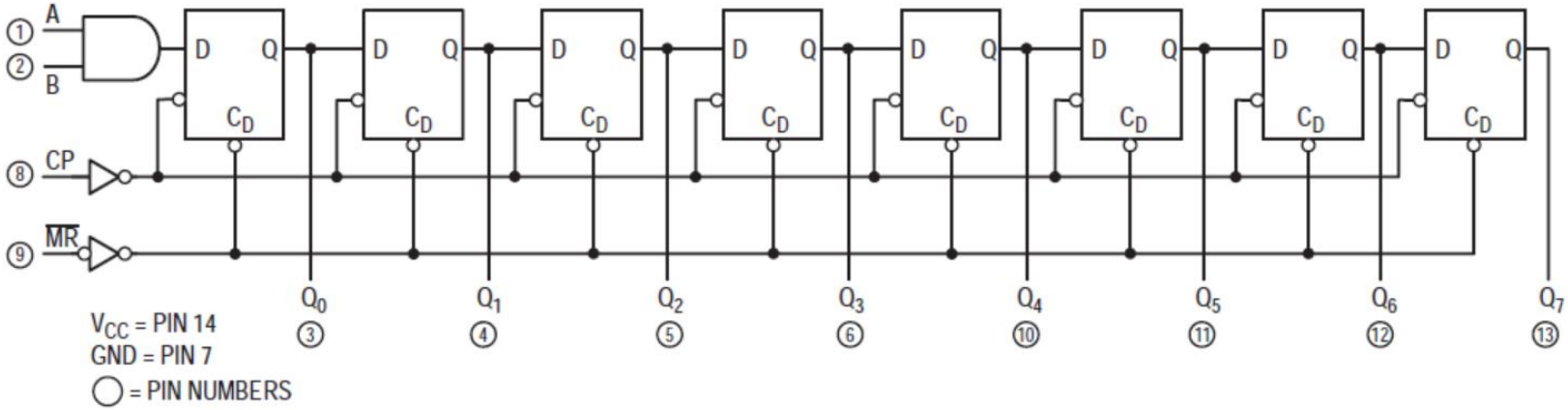
串入并出移位寄存器

◎ SN74LS164

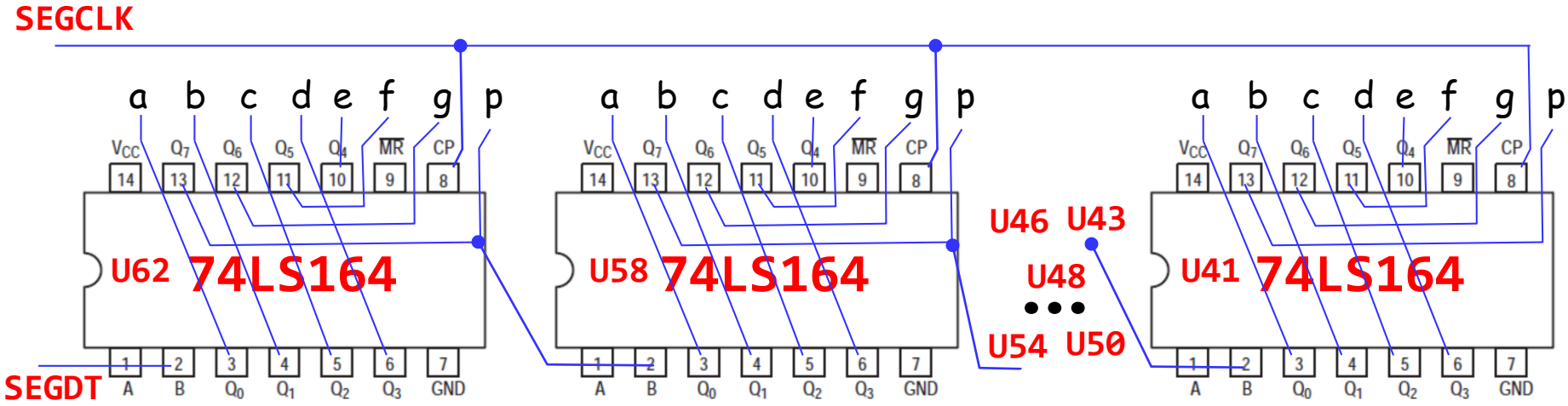
- 常用于串并转换
  - ⊙ 单片机、arduino等串行七段显示控制



OPERATING MODE	INPUTS			OUTPUTS	
	$\overline{MR}$	A	B	Q_0	Q_1–Q_7
Reset (Clear)	L	X	X	L	L – L
Shift	H	l	l	L	q_0 – q_6
	H	l	h	L	q_0 – q_6
	H	h	l	L	q_0 – q_6
	H	h	h	H	q_0 – q_6



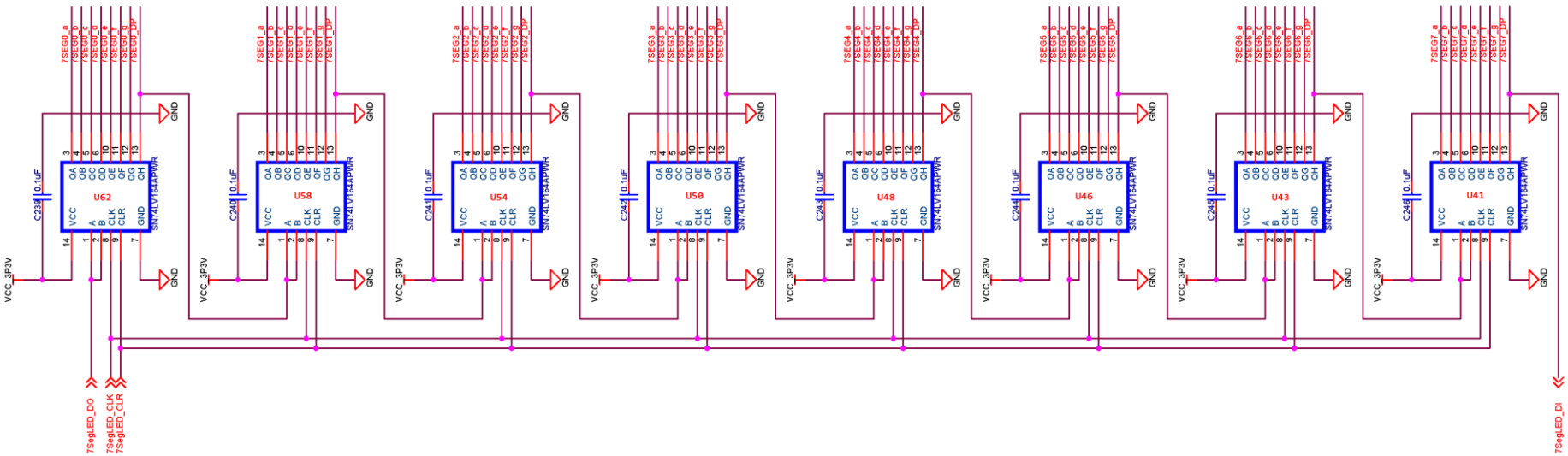
SN74LS164同步串行传输扩展



通过P2S模块输出：

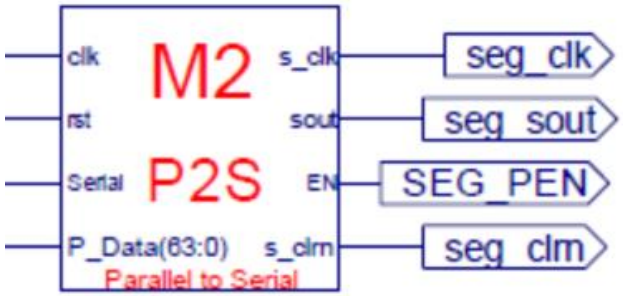
P_Data[63:0]=SEGMENT[63:0]

SN74LS164：SWORD 七段码接口

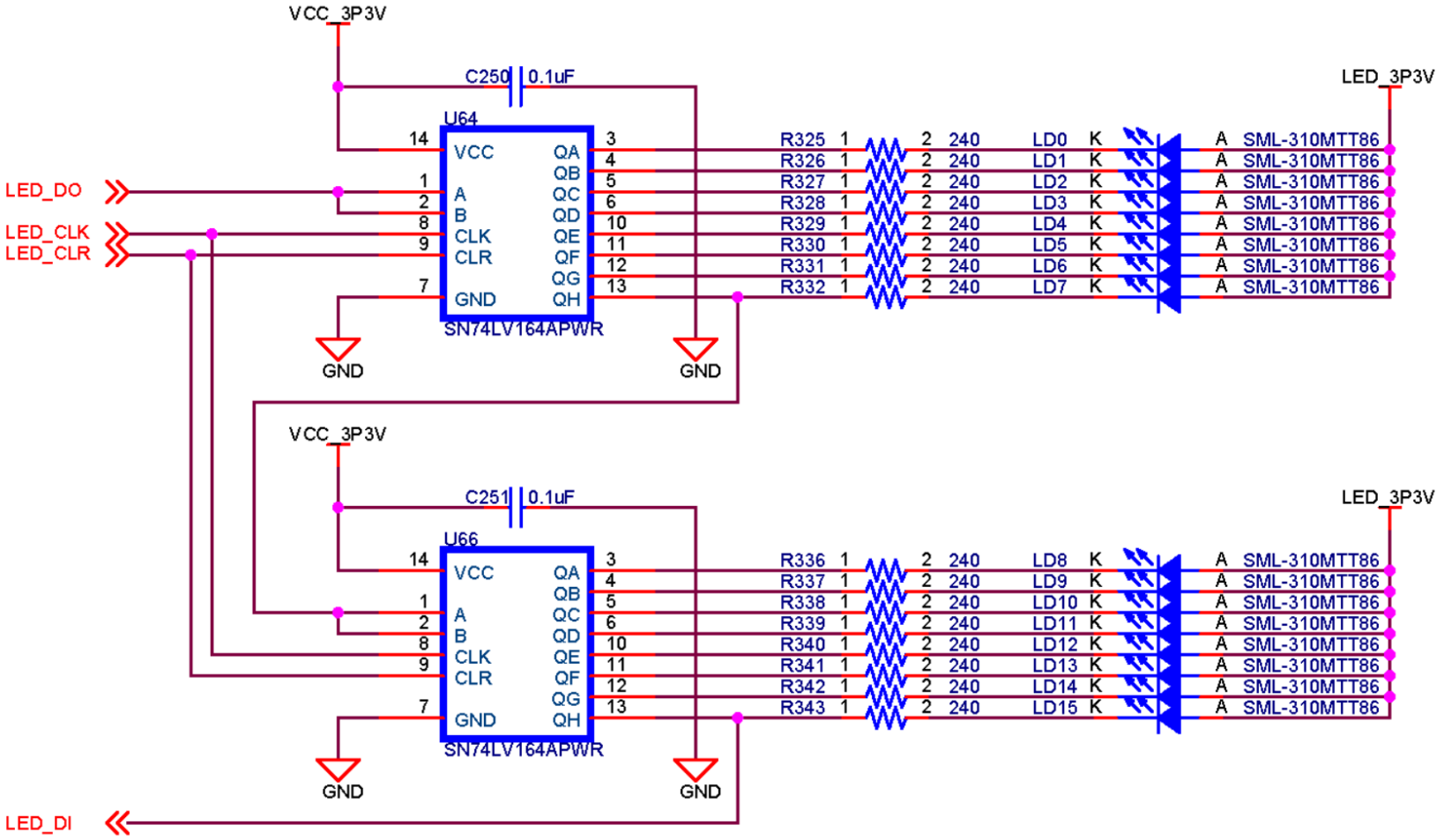


通过P2S模块输出：

P_Data[63:0]=SEGMENT[63:0]



SN74LS164：SWORD LED接口



Course Outline

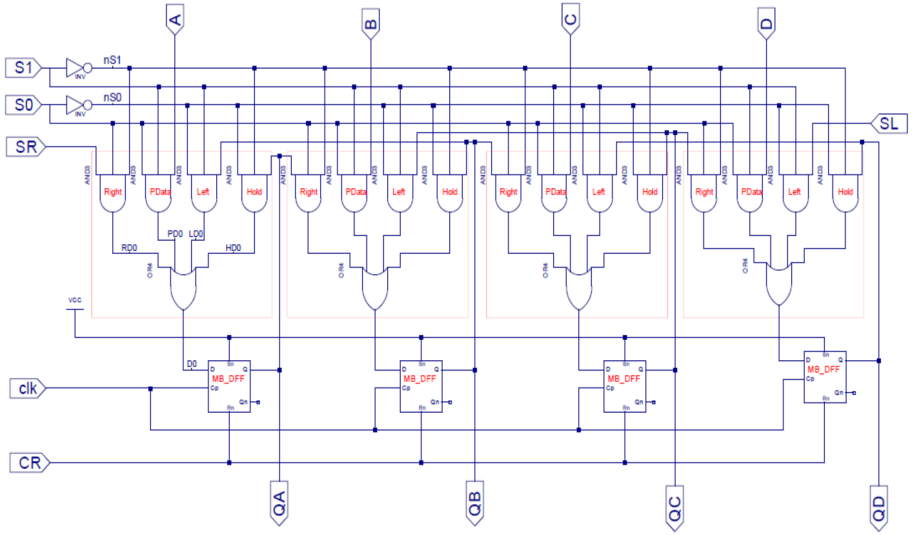
- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

设计工程一：Exp120-Shift

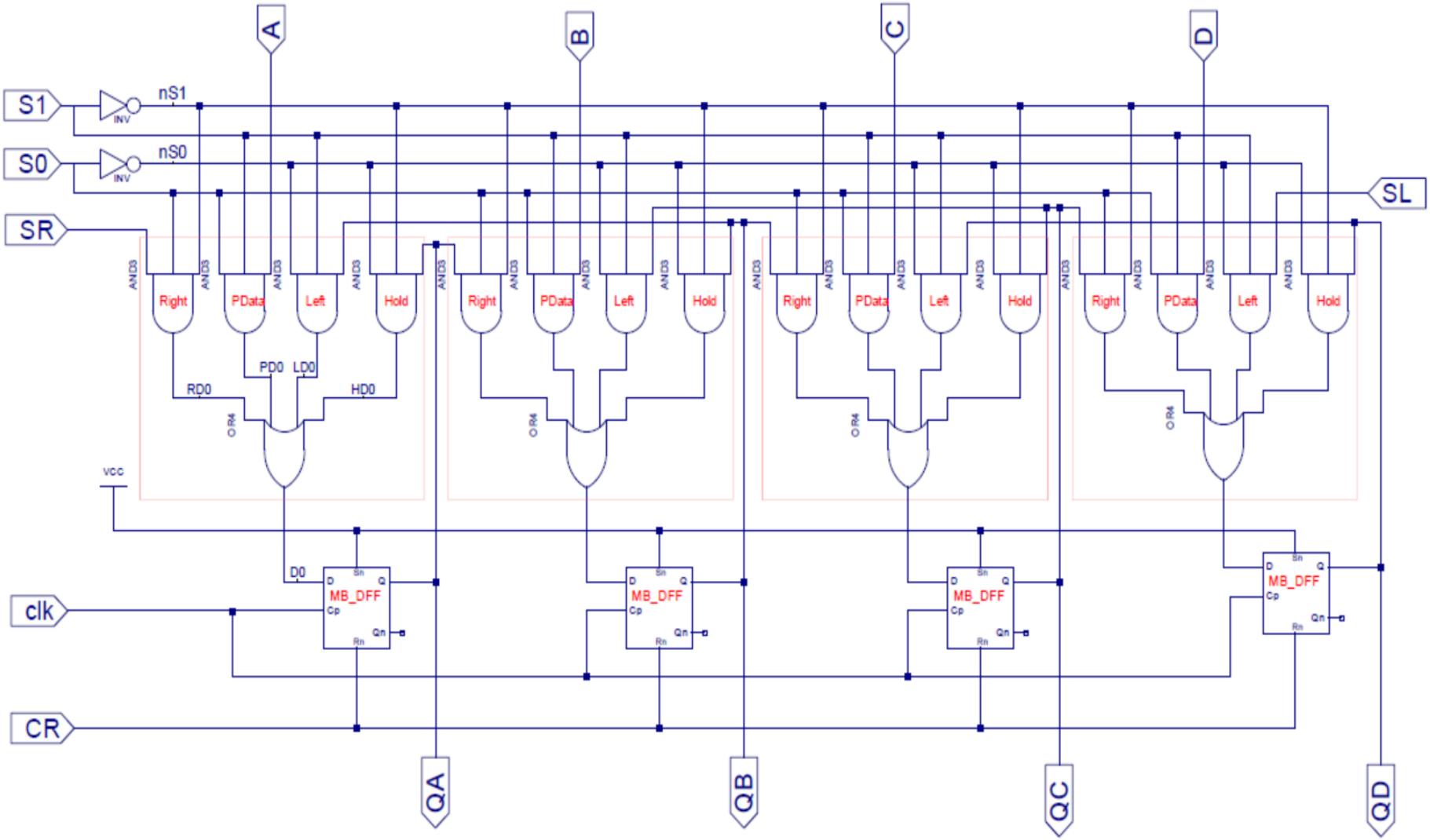
- **设计实现4位通用移位寄存器DM74LS194**
  - 调用实验九MB_DFF维持阻塞D触发器实现
  - 用原语和门级混合描述或原理图描述
- **设计实现32位通用移位寄存器**
  - 调用DM74LS194实现
  - 采用结构化行为描述或门级描述
- **集成移位寄存器到实验十一的“混合计算器”**
  - 修改实验十一顶层模块为：Top_shift
  - 其余功能不变，新增功能：
    - 32位通用移位寄存器
    - 寄存器输出显示用通道5(实验11的Q_B显示通道)

设计要点

- **新建工程：Exp12-Shift**
- **设计实现74LS194通用移位寄器存器**
  - 建议模块名：DM74LS194.v
  - 调用实验九MB_DFF 维持阻塞D触发器
  - 用Verilog HDL结构描述设计实现
    - 根据逻辑电路图用原语和门级混合描述



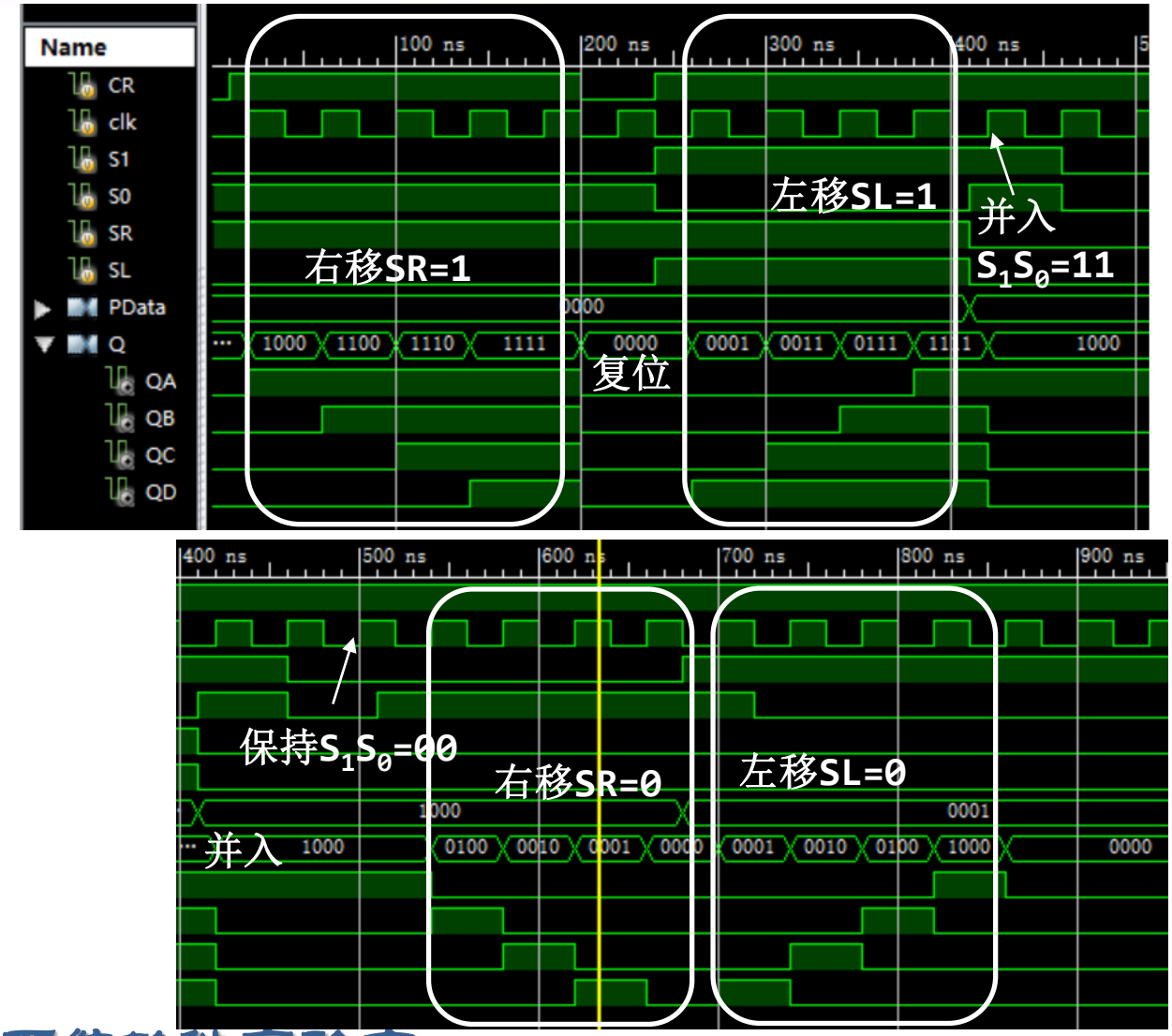
DM74LS194逻辑原理图：参考



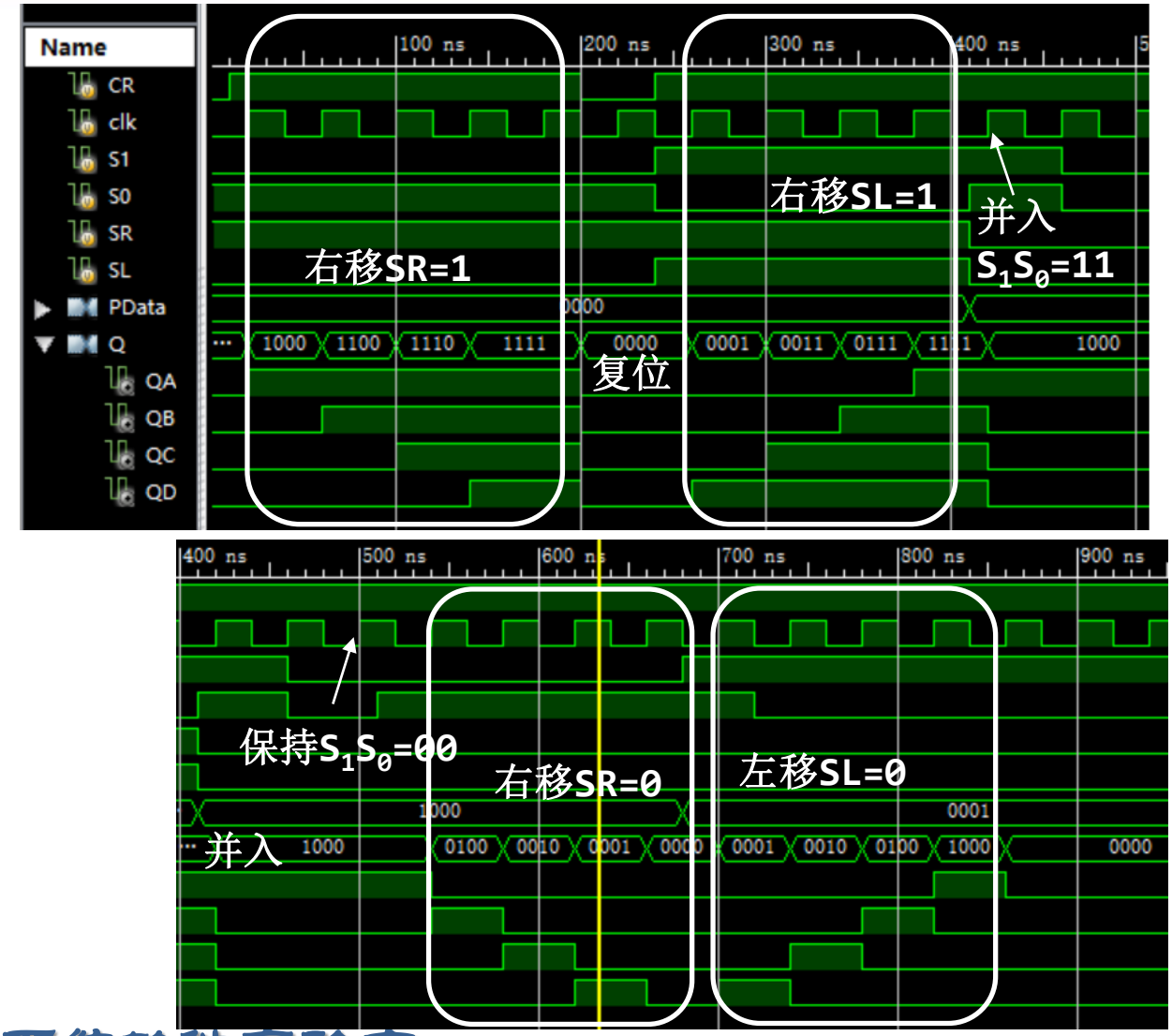
DM74LS194移位寄存器激励与仿真

◎ 激励要点

- 复位
- 右移SR=1
- 左移SL=1
- 并入、保持
- 右移SR=0
- 左移SL=0
- ……

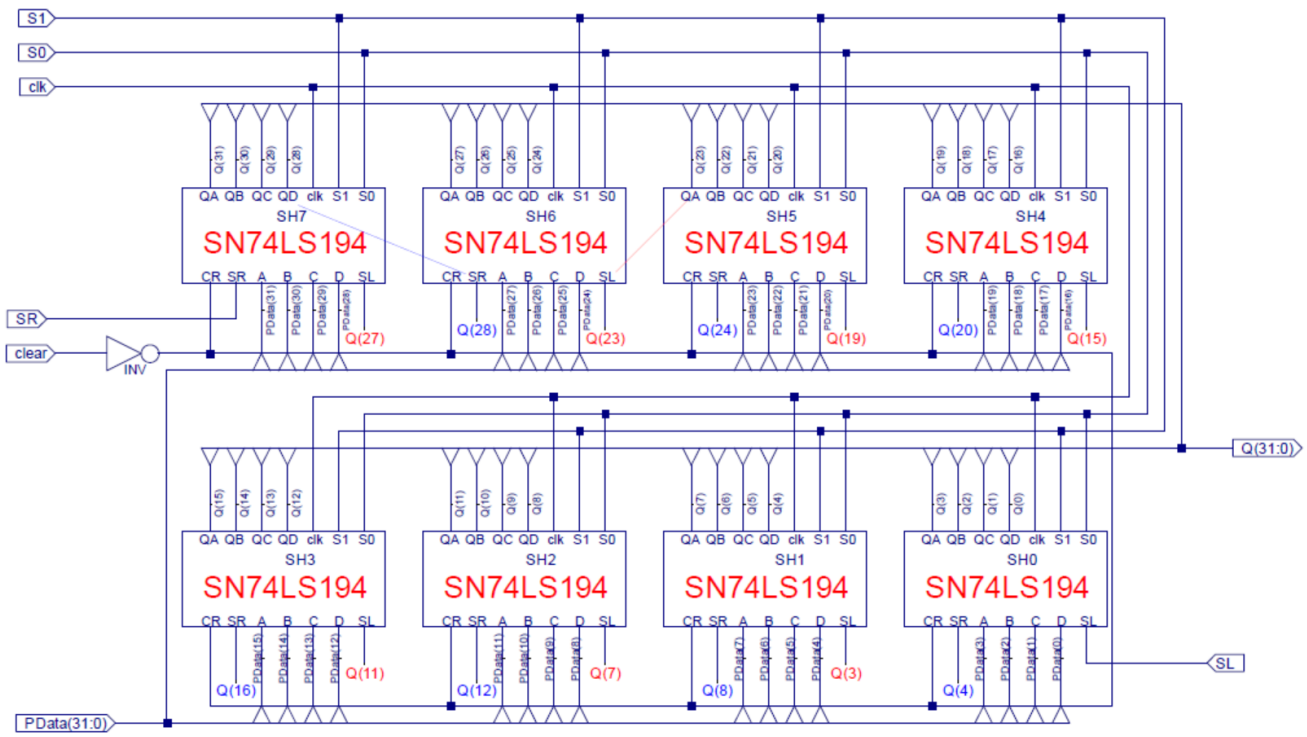


DM74LS194移位寄存器激励与仿真



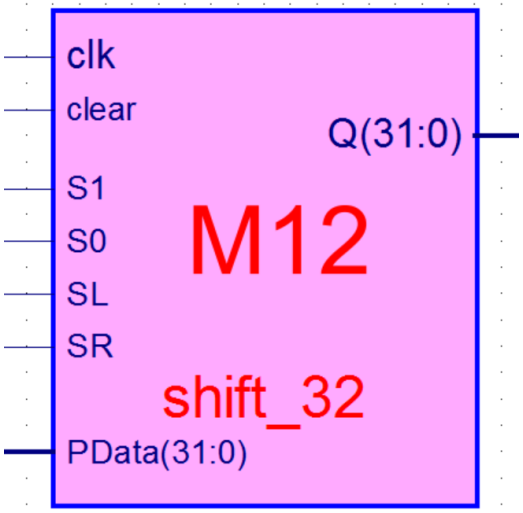
◎ 设计实现32位通用移位寄存器

- 调用DM74LS194实现，模块名Shift_32.v
  - ⊙ 门级描述，扩展4位通用移位寄存器
  - ⊙ 也可采用结构化行为描述
- 参考逻辑图



32位通用移位寄存器激励要点

- **激励要点**
  - 选择特征数据
    - 用独热码：One-hot code
      - 起始值=80000000H/00000001H
  - 右移32个周期，观察74LS194之间的连接
    - 复位 SR=1一个同期后SR=0
  - 并入PData=AAAAAAAAH→55555555H → 00000000H
  - 左移32个周期，观察74LS194之间的连接
    - SL=1一个同期后SL=0
- **仿真通过后制作逻辑符号**



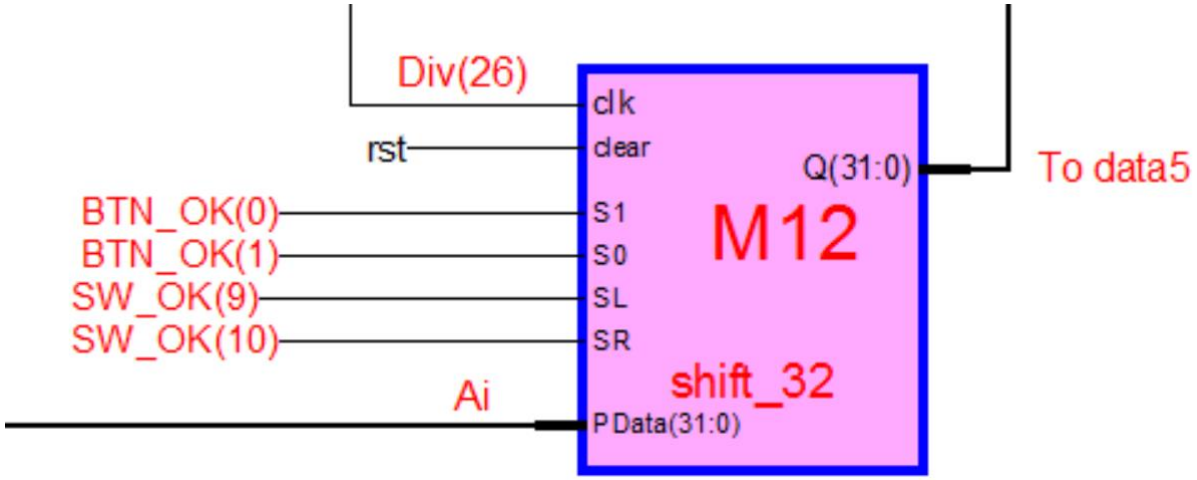
32位通用移位寄存器仿真图

?

集成混合计算器：增加移位功能

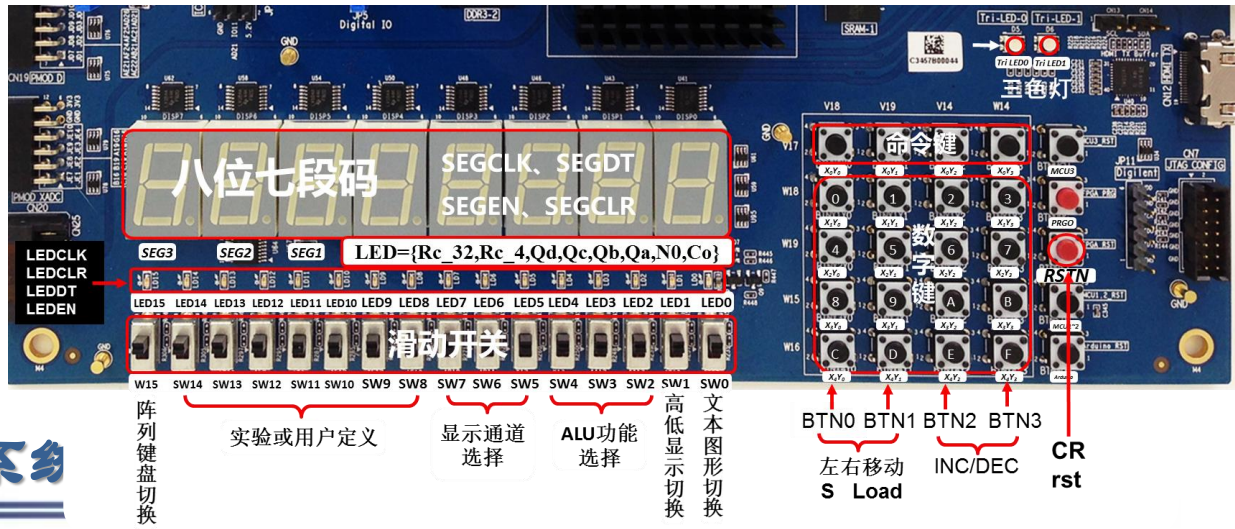
◎集成Calculation

- 复制实验十一的顶层模块，并改名为：Calculation_Exp120.sch
  - ⊙集成移位寄存器，命名M12。
- 接口分配
  - ⊙输入：clk=Div(26)；clear=rst(M2输出)；Pdata=Ai(M4输出)；
  - ⊙S_1S_0=BTN(1:0)；{SL,SR}=SW_OK(10:9)；
  - ⊙输出:Q→显示通道5(data5, 删除实验十一Q_B显示)

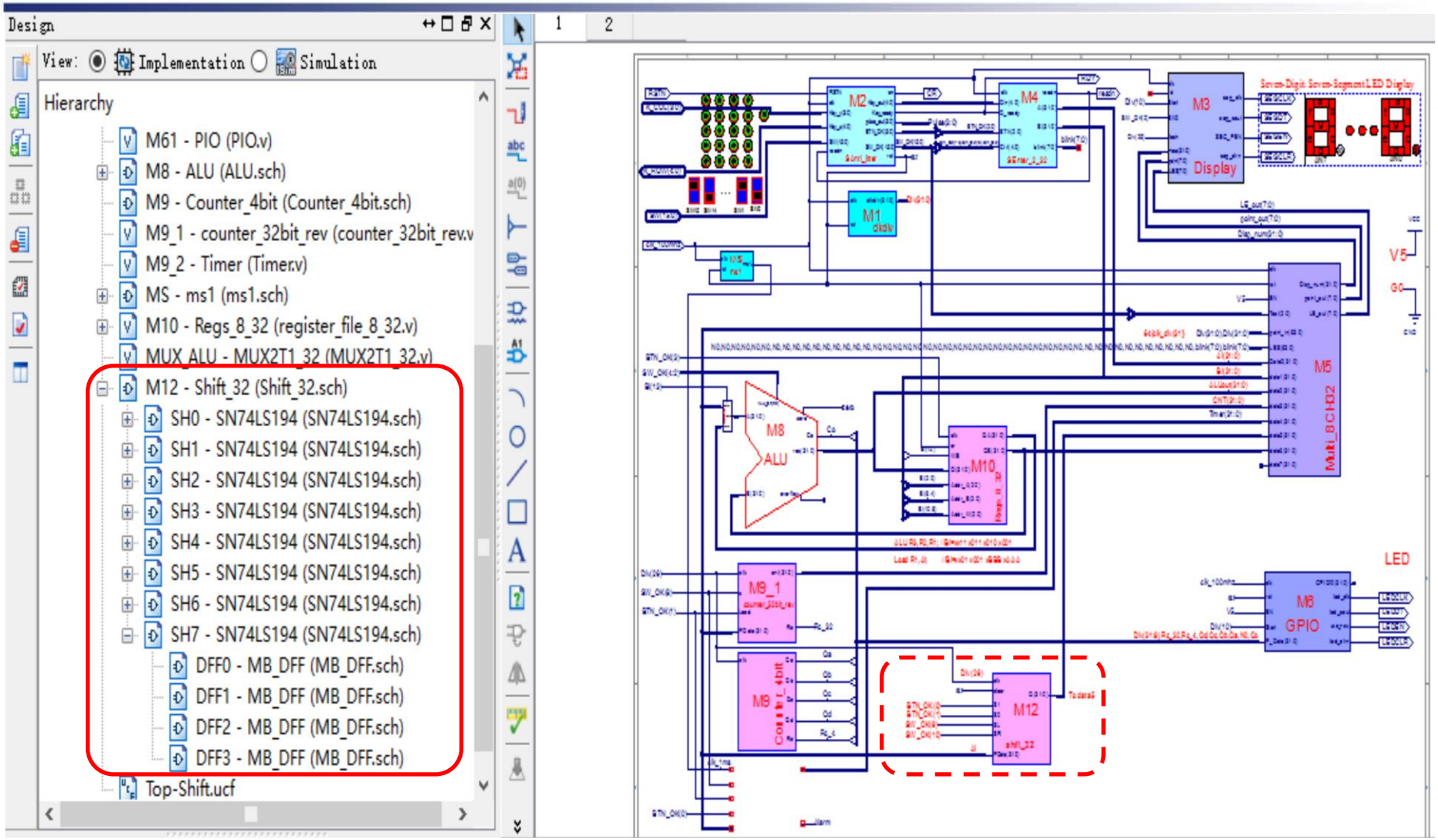


物理验证

- 输入
  - SW[7:5]=通道选择
    - =000: 修改被加数、=001: 修改加数、=010: ALU输出、=011:32位计数输出、=100:寄存器堆Q_A输出、=101:32位寄存器输出、**=111:移位输出**
  - SW[1]= 高低16位数据选择
  - BTN[1:0]={S1,S0}
  - SW[4:2]=ALU功能控制
- **输出:** {a~g，p }= SEGMENT，AN=AN，
  LED ={**Rc_32,Rc_4,Qd,Qc,Qb,Qa,N0,Co**}
- 同实验十一



实验十二的顶层结构


Design
View: Implementation Simulation
Hierarchy
M61 - PIO (PIO.v)
M8 - ALU (ALU.sch)
M9 - Counter_4bit (Counter_4bit.sch)
M9_1 - counter_32bit_rev (counter_32bit_rev.v
M9_2 - Timer (Timer.v)
MS - ms1 (ms1.sch)
M10 - Regs_8_32 (register_file_8_32.v)
MUX_ALU - MUX2T1_32 (MUX2T1_32.v)
M12 - Shift_32 (Shift_32.sch)
SH0 - SN74LS194 (SN74LS194.sch)
SH1 - SN74LS194 (SN74LS194.sch)
SH2 - SN74LS194 (SN74LS194.sch)
SH3 - SN74LS194 (SN74LS194.sch)
SH4 - SN74LS194 (SN74LS194.sch)
SH5 - SN74LS194 (SN74LS194.sch)
SH6 - SN74LS194 (SN74LS194.sch)
SH7 - SN74LS194 (SN74LS194.sch)
DFF0 - MB_DFF (MB_DFF.sch)
DFF1 - MB_DFF (MB_DFF.sch)
DFF2 - MB_DFF (MB_DFF.sch)
DFF3 - MB_DFF (MB_DFF.sch)
Top-Shift.ucf
1
2
M2
M4
M3
Display
M1
M5
Multi_BCH32
M8
ALU
M10
M9_1
M9
Counter_4bit
M12
M6
GPIO
LED

输入设备功能定义

开关定义	=0	=1	备注
SW[0]	未用		
SW[1]	32位二进制高16位	32位二进制低16位	
SW[4:2]	ALU功能选择		参考ALU功能表
SW[7:5]	通道选择 =000 =001 =010 =011 **=100** **=101** =110 =111	 通道0 通道1 通道2 通道3 **通道4** **通道5** 通道6 通道7	 **A*i*** **B*i*** **RES(ALU_Out)** **Cnt** **寄存器A输出QA** **移位寄存器输出Q**

按键定义	=0	=1	备注
Button[0]	左移	右移	移动方向控制
Button[1]		正脉冲移动	
Button[2]		正脉冲输入修改	递增修改
Button[3]			长按复位

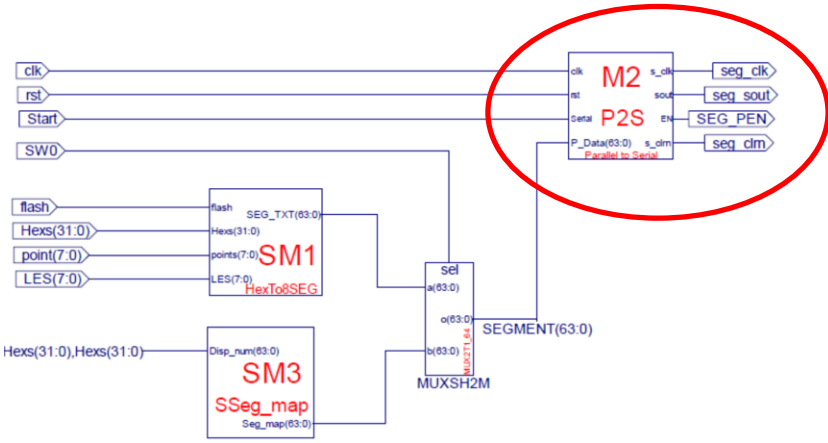
移位寄存器功能控制

开关定义	=0	=1	备注
BTN[1:0]=S1S0	=00 =01 =10 =11		保持 右移 左移 并入
SW[10:9]={SL,SR}	串行移入0	串行移入1	

设计工程二：通用并-串转换器-P2S

课后实践

- **设计*n*位双向移位寄存器**
  - 用HDL结构化行为描述
  - 宽度(位数)*n*可变
  - 时序仿真检测
- **设计并串转换模块:P2S**
  - 调用上述移位寄存器
  - 串行输出适应：高位或低位在先输出(大小端模式)
- **集成P2S模块替换实验环境中的P2S核**
  - 替换实验七工程四中的LED和七段码输出模块
    - LEDP2S.ngc、P2S.ngc




```
LED_P2S #(.DATA_BITS(16),.DATA_COUNT_BITS(4))
LED_P2S( clk, rst, Start,
         LED,
         led_clk,
         led_clrn,
         led_sout,
         LED_PEN
         );
```

移位寄存器器应用：P2S模块设计

```
1   module          P2S(input wire clk,                          //parallel to serial
2                       input wire rst,
3                       input wire Start,
4                       input wire[DATA_BITS-1:0] PData,
5                       output wire s_clk,
6                       output wire s_clrn,
7                       output wire sout,
8                       output reg  EN
9                       );
10
11  parameter
12      DATA_BITS = 16,                                          // data length
13      DATA_COUNT_BITS = 4,                                     // data shift bits
14      DIR = 0;                                                 // Shift direction =0左移
15
16  wire S1,S0,SL,SR;
17  wire [DATA_BITS:0] D,Q;
18  reg [1:0]Go = 00, S = 00;
19
20      assign {SR,SL} = 2'b11;                                  //左移右移初值=1，用于传输标志
21      assign {S1,S0} = DIR ? {S[0],S[1]} : S;                  //调整移位方向
22      assign D       = DIR ? {1'b0,PData} : {PData,1'b0};              //置移位末端标志"0"
23      wire finish    = DIR ? &Q[DATA_BITS:1] : &Q[DATA_BITS-1:0];//移位传输结束：全"1"
24      assign sout    = DIR ? Q[0] : Q[DATA_BITS];              //串行输出：右移输出Q[0]
25
```

```
26    SHIFT64    #(.DATA_BITS(DATA_BITS))  //调用移位寄存器，宽度=DATA_BITS
27               PTOS(.clk(clk),
28                    .SR(SR), .SL(SL),
29                    .S1(S1), .S0(S0),
30                    .D(D), .Q(Q)
31                    );
32
33    always @(posedge clk )                          //采样Start上升沿
34       Go <= {Go[0],Start};
35    assign shift    =(Go==2'b01)? 1:0;              //移位启动采样
36
37    always @(posedge clk or posedge rst)begin
38       if(rst) begin EN = 1; S = 2'b11; end
39       else begin
40          if(shift) begin     EN = 0;   S = 2'b11; end //并行置入
41          else begin
42             if(!finish)begin EN = 0; S = 2'b10; end   //启动移位传输
43             else begin       EN = 1; S = 2'b00; end   //结束移位
44          end
45       end
46    end
47    assign s_clk = finish | clk;                    //禁止74LS164时钟
48    assign s_clrn = 1;
49
50 endmodule
```

参数传入

N位双向并入串出移位寄存器：宽度=DATA_BITS

启动传输结束控制块

同学们：每次做完实验请整理好实验台，放好仪器，理清桌面。


Thank you!